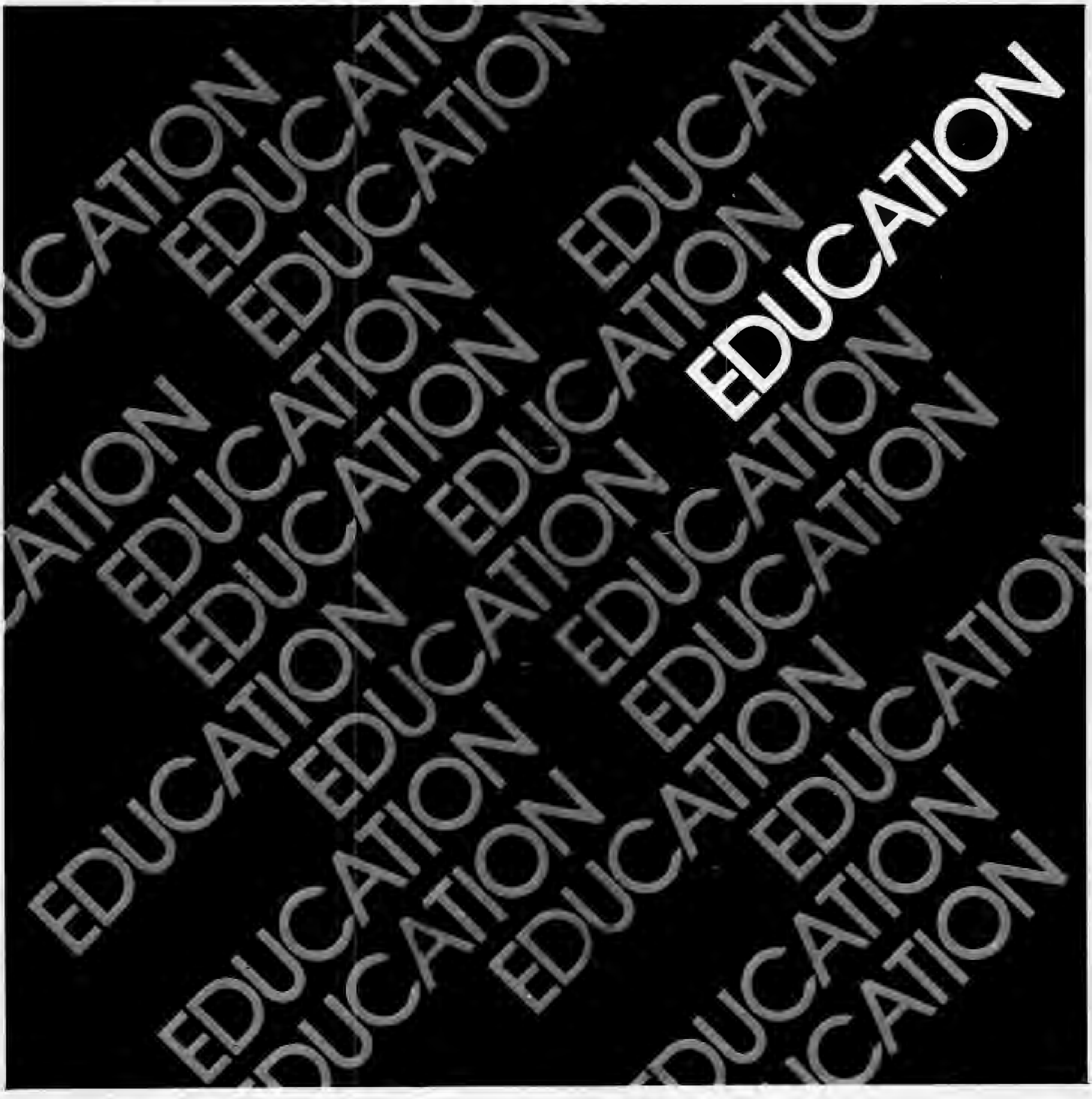
digital



"© Copyright 1978 by Digital Equipment Corporation"

RT-11 USER
HANDOUTS

October 25, 1978

RT-11 USER

COURSE OUTLINE

MONDAY	OVERVIEW OF RT-11 MONITOR COMMANDS UTILITY FUNCTIONS CONCISE COMMAND LANGUAGE
TUESDAY	PROGRAM DEVELOPMENT EDITOR MACRO FORTRAN LINK
WEDNESDAY	PROGRAM DEBUGGING TOOLS LIBRARIES INDIRECT COMMAND FILES RT-11 FILE STRUCTURE
THURSDAY	OVERLAYS FOREGROUND-BACKGROUND MONITOR BASIC
FRIDAY	SYSTEM GENERATION BATCH PROCESSES

RT-11 USER

READING ASSIGNMENTS

MONDAY:	INTRO. TO RT-11	CHS. 1-4,7
	SYS. USER GUIDE	CHS. 1-3 CH4.4 (SKIM CMDS)
TUESDAY:	INTRO. TO RT-11	CHS. 5,6,8,9,11-13
	SYS. USER GUIDE	CH.4 (SKIM CMDS) CH.5
WEDNESDAY:	INTRO. TO RT-11	CHS. 14,16
	SYS. USER GUIDE	CHS. 4.3,12,16
THURSDAY:	INTRO. TO RT-11	CHS. 10,15
	SYS. USER GUIDE	CH. 11.6
FRIDAY:	SYSGEN	CHS. 1-3
	SYS. USER GUIDE	CHS.17,18 APP.A

RT-11 MINIMUM REQUIREMENTS

RANDOM ACCESS DEVICE

- DISKS - RP02, RP03, RK06, RK05, RS11,
  RS03, RS04
- DISKETTE - RX01 (FLOPPY)
- DECTAPE - TU56

CONSOLE TERMINAL

single job
FOR SJ MONITOR - 8K WD. MEMORY

foreground/background
FOR FB MONITOR - 16K WD. MEMORY
KW11 LINE FREQUENCY CLOCK

extended memory.
FOR XM MONITOR - 24K WD. MEMORY
KT11 MEMORY MANAGEMENT UNIT

RT-11 MONITOR

PROVIDES THE USER WITH

.CONSOLE TERMINAL SERVICE

.SYSTEM COMMANDS

.PROGRAM REQUEST SERVICES

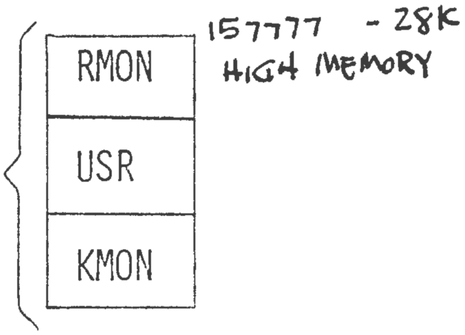
.FILE SUPPORT

.ACCESS TO DEVICE HANDLERS

EXISTS AS THREE SEPARATE SECTIONS IN MEMORY, THEY ARE

RESIDENT MONITOR	RMON
USER SERVICE ROUTINES	USR
KEYBOARD MONITOR	KMON

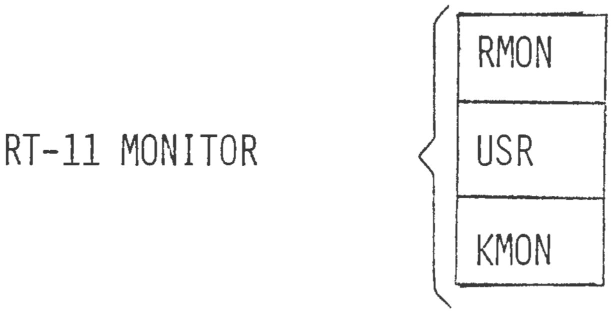
RT-11 MONITOR



RESIDENT MONITOR (RMON)

- PROGRAMMED REQUESTS (PR'S) FOR ALL SERVICES OF RT-11
- CONSOLE TERMINAL SERVICE
- ERROR PROCESSOR
- EMT PROCESSOR
- SYSTEM TABLES
- SYSTEM DEVICE HANDLER

RMON IS THE CORE RESIDENT PART OF RT-11



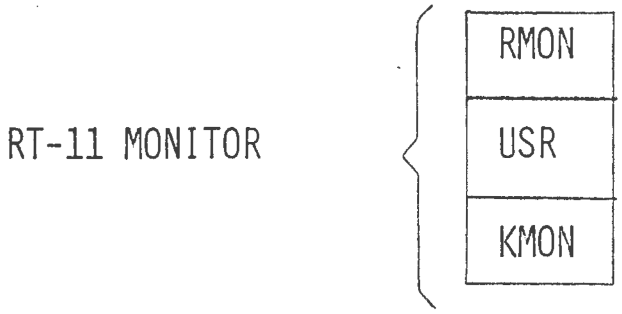
USER SERVICE ROUTINES (USR)

- PROVIDES SUPPORT FOR RT-11 FILE STRUCTURE
  - A) LOADS DEVICE HANDLERS
  - B) OPENS FILES
  - C) READS OR WRITES FILES
  - D) DELETES, RENAMES FILES
  - E) CREATES NEW FILES

- USES THE COMMAND STRING INTERPRETER (CSI) TO DECODE DEVICE AND FILE I/O INFORMATION. CSI IS PART OF THE USR.

- SWAPS IN AND OUT TO ALLOW THE USER MAXIMUM CORE AVAILABILITY. SWAP AREA ~~PART OF MONITOR~~ FILE ON DISK. SWAP.SYS

USR & KMON ARE WRITTEN TO AND READ FROM SWAP.SYS



KEYBOARD MONITOR (KMON)

- ALLOWS THE USER TO:

A) ASSIGN LOGICAL NAMES TO DEVICES

B) RUN PROGRAMS

C) LOAD DEVICE HANDLERS

D) CONTROL F/B OPERATIONS

E) MODIFY MEMORY LOCATIONS

- PROVIDES COMMUNICATION BETWEEN THE USER CONSOLE AND RT-11 VIA THE SYSTEM COMMANDS

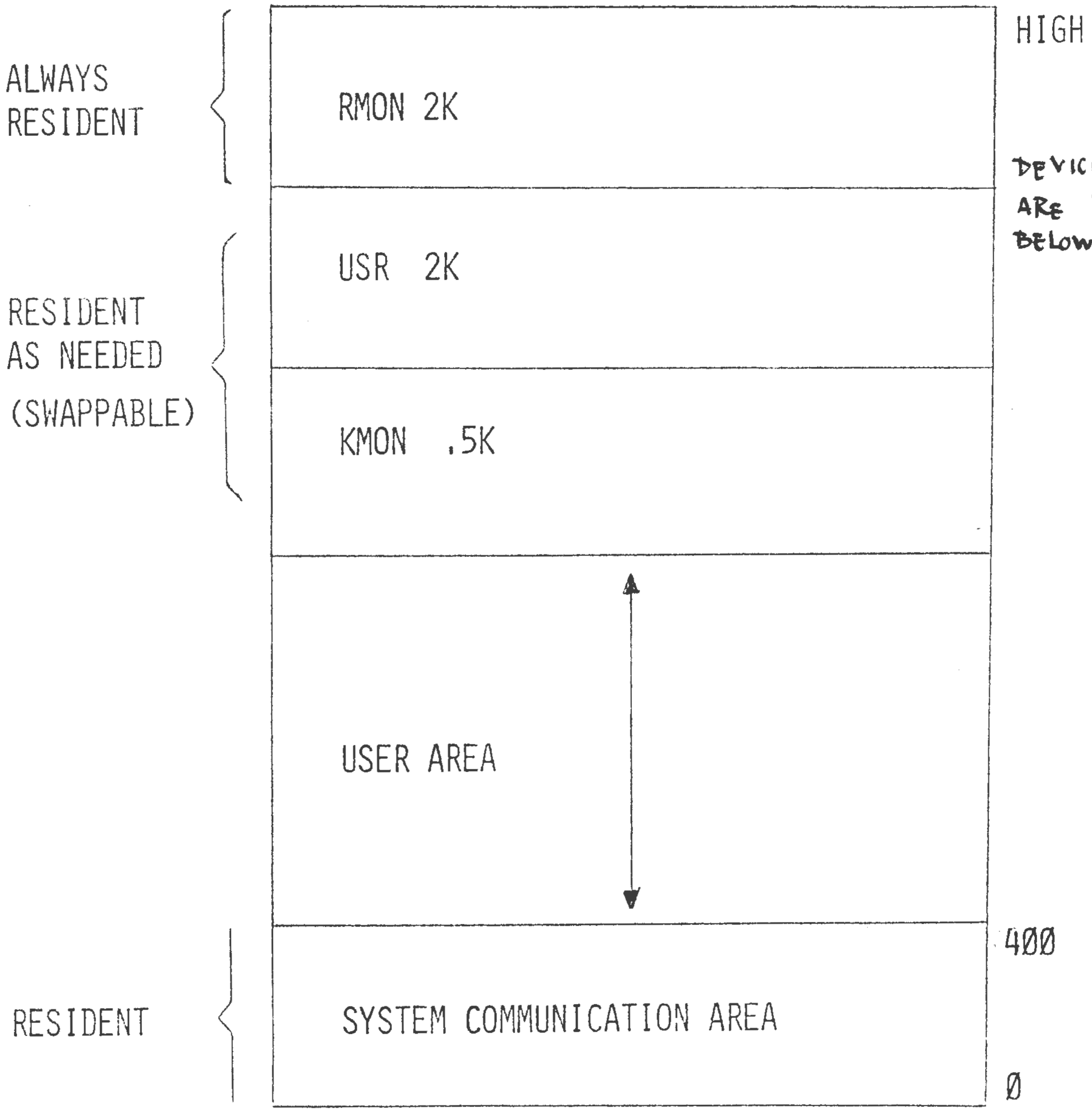
- PROMPTS THE CONSOLE WITH A PERIOD(.)

- UNDER THE F/B MONITOR, KMON (WHEN RUNNING) IS THE BACKGROUND JOB.

- SWAPS TO ALLOW MAXIMUM CORE AVAILABILITY. - SWAP AREA PART OF MONITOR FILE ON DISK.

SINGLE JOB MONITOR

CORE LAYOUT



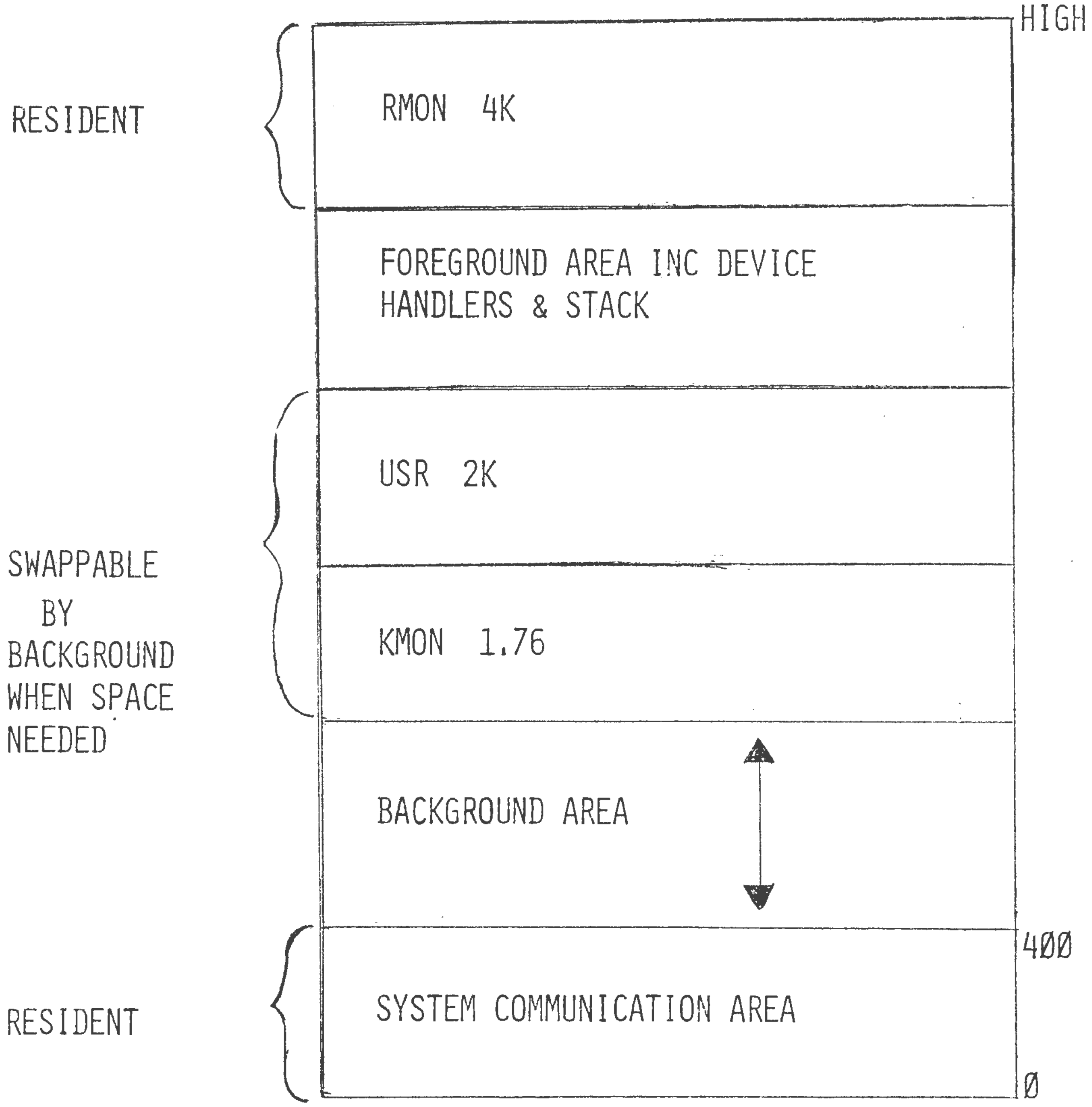
DEVICE HANDLERS ARE LOADED BELOW RMON

USER PROGRAMS CAN GO ALL THE WAY UP TO RMON

1000-400 STACK

FOREGROUND/BACKGROUND MONITOR

CORE LAYOUT
FOREGROUND LOADED


HIGH
RESIDENT
RMON 4K
FOREGROUND AREA INC DEVICE HANDLERS & STACK
USR 2K
SWAPPABLE BY BACKGROUND WHEN SPACE NEEDED
KMON 1.76
BACKGROUND AREA
400
RESIDENT
SYSTEM COMMUNICATION AREA
0

```
SHOW
TT  (RESIDENT)
RK  (RESIDENT)
  RKO   =  SY , DK
<FREE>
<FREE>
<FREE>
<FREE>
<FREE>
DX
DT
MT
<FREE>
<FREE>
LP
BA

SHOW DEVICES

DEVICE       STATUS            VECTOR
-------------------------------------
  DX       INSTALLED           000264
  DT       INSTALLED           000214
  RK       RESIDENT            000220
  LP       INSTALLED           000200
  CR       NOT INSTALLED       000230
  MT       INSTALLED           000224
  NL       NOT INSTALLED       000000
  PC       NOT INSTALLED       000070 000074
  CT       NOT INSTALLED       000260


.SHOW CONFIGURATION

RT-11FB    V03B-00

BOOTED FROM RKO:
RESIDENT MONTOR BASE IS 140026
USR IS SET SWAP
TT IS SET NOQUIET
INDIRECT FILE ABORT LEVEL IS ERROR
INDIRECT FILE NESTINS DEPTH IS 3

PDP 11/04 PROCESSOR
PARITY MEMORY
60 CYCLE SYSTEM CLOCK

NO SYSGEN OPTIONS ENABLED
```

```
.DIRECTORY DT0:

VTMAC .MAC     7 01-Aug-77      DEMOFG.MAC     5 01-Aug-77
DT    .MAC     7 14-Aug-77      FSM   .MAC    31 14-Aug-77
DEMOFG.SAV     7 07-Dec-77      DEMOFG.REL     8 07-Dec-77
DEMOFG.OBJ     1 08-Dec-77      DT    .SYS     2 14-Aug-77
DTX   .SYS     2 14-Aug-77      DM    .SYS     4 14-Aug-77
DM    .MAC    19 14-Aug-77
 11 Files, 93 Blocks
 471 Free blocks
```

```
.DIRECTORY/BRIEF DT0:

VTMAC .MAC    DEMOFG.MAC    DT    .MAC    FSM   .MAC    DEMOFG.SAV
DEMOFG.REL    DEMOFG.OBJ    DT    .SYS    DTX   .SYS    DM    .SYS
DM    .MAC
 11 Files, 93 Blocks
 471 Free blocks
```

```
.DIRECTORY/BLOCKS DT0:

VTMAC .MAC     7 01-Aug-77    14      DEMOFG.MAC     5 01-Aug-77    21
DT    .MAC     7 14-Aug-77    26      FSM   .MAC    31 14-Aug-77    33
DEMOFG.SAV     7 07-Dec-77    64      DEMOFG.REL     8 07-Dec-77    71
DEMOFG.OBJ     1 08-Dec-77    79      DT    .SYS     2 14-Aug-77    80
DTX   .SYS     2 14-Aug-77    82      DM    .SYS     4 14-Aug-77    84
DM    .MAC    19 14-Aug-77    88
 11 Files, 93 Blocks
 471 Free blocks

.
```

```
DK:SAMPLE.DMP
BLOCK NUMBER  00000
000/ 044124 051511 044440 020123 020101 040523 050115 042514 *THIS IS A SAMPLE*
020/ 005015 052504 050115 043040 046111 027105 006456 030412 *..DUMP FILE....1*
040/ 031462 032464 033466 034470 006460 040412 041502 042504 *234567890..ABCDE*
060/ 043506 044510 045512 006514 046412 047516 050520 051522 *FGHIJKL..MNOPQRS*
100/ 052524 053526 054530 006532 000012 000000 000000 000000 *TUVwXYZ.........*
120/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
140/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
160/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
200/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
220/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
240/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
260/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
300/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
320/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
340/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
360/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
400/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
420/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
440/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
460/ 000000 000000 000000 000000 000000 000000 000000 000000 *................*
```

```
DK:SAMPLE.DMP/B
BLOCK NUMBER  00000
000/ 124 110 111 123 040 111 123 040 101 040 123 101 115 120 114 105
     T   H   I   S       I   S       A       S   A   M   P   L   E
020/ 015 012 104 125 115 120 040 106 111 114 105 056 056 015 012 061
     .   .   D   U   M   P       F   I   L   E   .   .   .   .   1
040/ 062 063 064 065 066 067 070 071 060 015 012 101 102 103 104 105
     2   3   4   5   6   7   8   9   0   .   .   A   B   C   D   E
060/ 106 107 110 111 112 113 114 015 012 115 116 117 120 121 122 123
     F   G   H   I   J   K   L   .   .   M   N   O   P   Q   R   S
100/ 124 125 126 127 130 131 132 015 012 000 000 000 000 000 000 000
     T   U   V   w   X   Y   Z   .   .   .   .   .   .   .   .   .
120/ 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
     .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
140/ 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
     .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
160/ 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
     .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
200/ 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
     .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
220/ 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
     .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
240/ 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
     .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .
260/ 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000 000
```

TO CREATE NEW SYSTEM DISK ON DX1:

. INITIALIZE DX1: INITIALIZE/BAD DX1:

. COPY FILES TO DX1: JUST THE APPRIATE FILES

. COPY/BOOT DX1:DXMNSJ.SYS DX1
DX1: DXMNFB.SYS DX1:
WILL BRING UP F/B MONITOR

TO DISCOVER AND MASK BAD BLOCKS ON RK0:

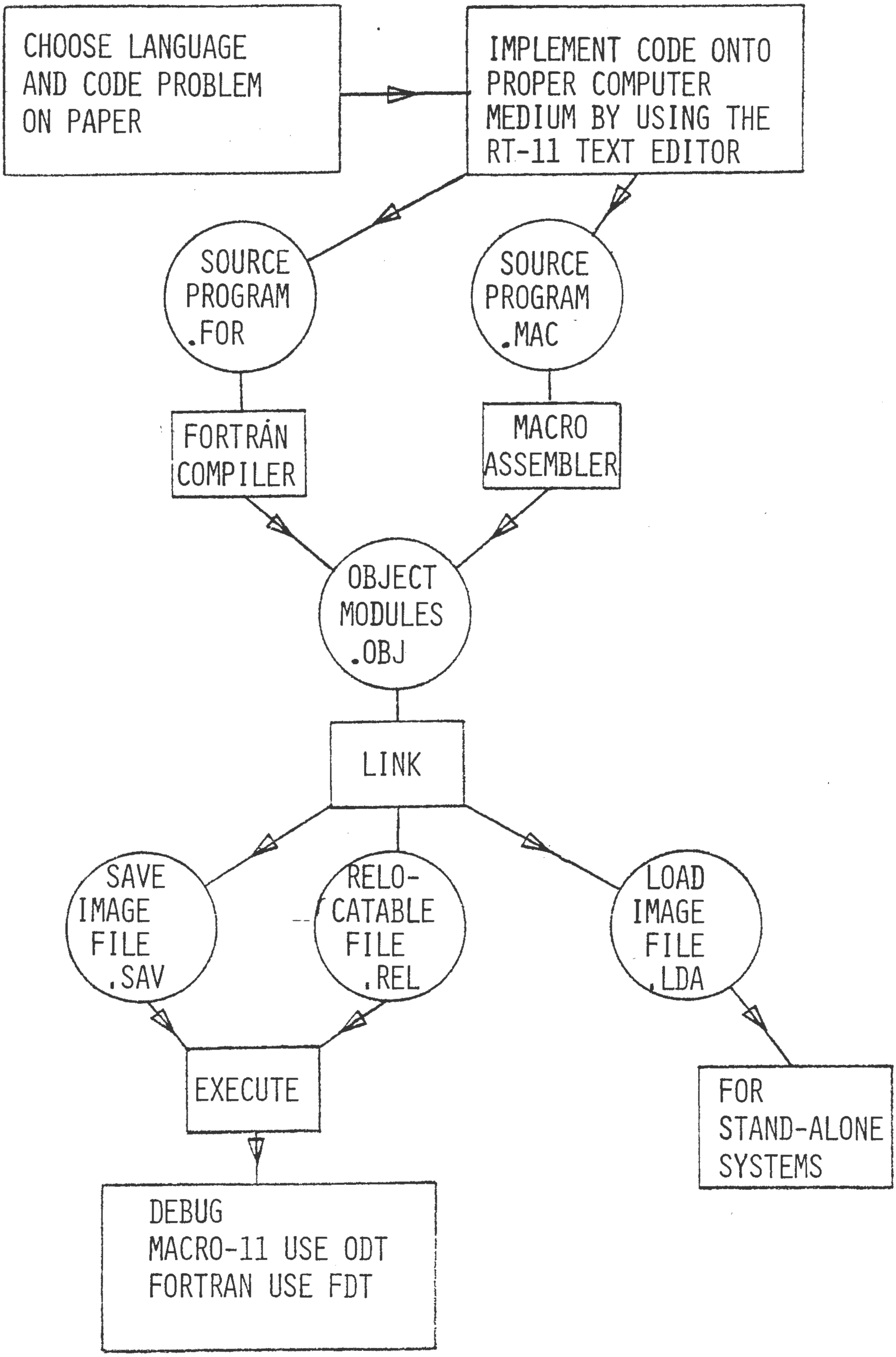
. DIRECTORY/BADBLOCKS/FILES RK0:

[RECOPY FILES AS NECESSARY AND DELETE THEM -
ASSUME BLOCK 37 BAD] .

. RUN DUP

* FILE.BAD = /C:37:1

RT-11 PROGRAM DEVELOPMENT



EDITOR SESSION

```
.EDIT/CREATE DEMO.MAC
*I      .MCALL .EXIT
START:  CLR     R0
        MOV     #1647.,R1
A1:     RAL\LA\OL       WRD
        ONC R9^U
        INC     R0
        DEC     R3
A2:     INC     R1
        VMT     A1
        .EXUT
WORD;;; .WORD   3762.
        .END START
$$
*B$$
*10J$$
*0L$$
        .MCALL .E*-3J0L$$
        .MCALL*4A$$
*V$$
        INC     R0
*-2AV$$
        MOV     #1647.,R1
*3L$$
        MOV     #1647.,R1
A1:     ROL     WRD
        INC     R0
*V$$
        MOV     #1647.,R1
*



*B$$
*G1647$$
*V$$
        MOV     #1647.,R1
*0L$$
        MOV     #1647*BG1647$0L$$
        MOV     #1647*-2DV$$
        MOV     #16.,R1
*GW$0L$$
A1:     ROL     W*IO$V$$
A1:     ROL     WORD
*GVMT$0L$$
        VMT*-3CBLT$V$$
        BLT     A1
*GEXUT$0L$$
        .EXUT*=CEXIT$V$$
        .EXIT
*GWORD$3C:$V$$
WORD:   .WORD   3762.
*
```

EDITOR SESSION

```
*B$$
*GINC$V$$
        INC     R0
*0L$$
        INC*0A0L$$
*I      BPL     A2
$$
*-AV$$
        BPL     A2
*4L$$
        BPL     A2
        INC     R0
        DEC     R3
A2:     INC     R1
*2AV$$
        DEC     R3
*K$$
*B/L$$
        .MCALL .EXIT
START:  CLR     R0
        MOV     #16.,R1
A1:     ROL     WORD
        BPL     A2
        INC     R0
A2:     INC     R1
        BLT     A1
        .EXIT
WORD:   .WORD   3762.
        .END START
*EX$$

.
```

MODIFICATION OF DEMO. MAC

```
.EDIT DEMO.MAC
```

CREATES TENTATIVE FILE DEMO.MAC
UPON EXIT RENAME DEMO.MAC TO DEMO.BAK
AND MAKES DEMO.MAC PERMANENT

```
*R$$
```

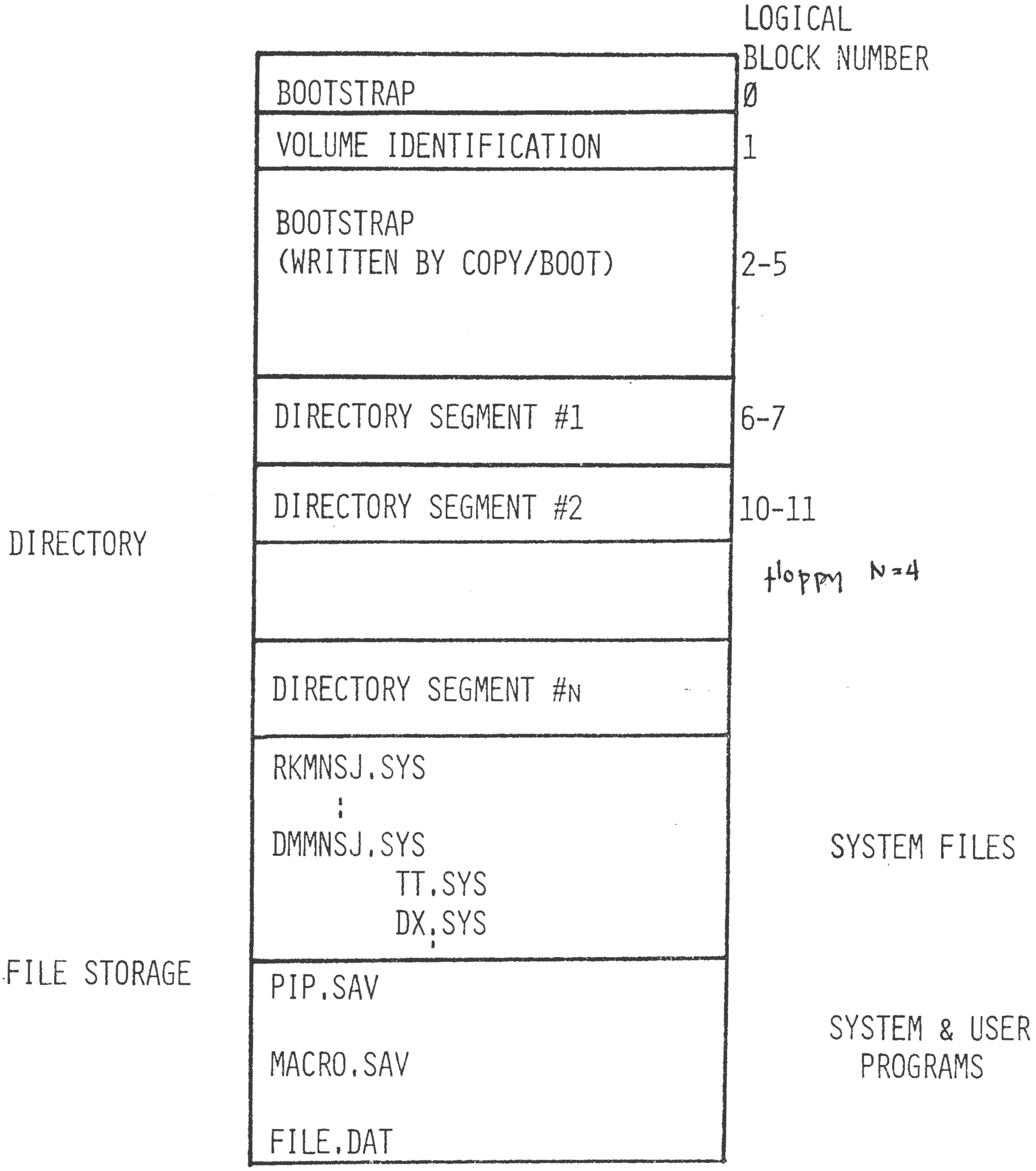
THE READ COMMAND READS ONE LOGICAL PAGE OF TEXT INTO THE TEXT BUFFER. A LOGICAL PAGE IS TERMINATED VIA:

1. A FORM FEED CHARACTER - INSERTED BY USER.
2. AN END OF FILE.

IF THE EDITOR'S BUFFER SPACE IS WITHIN 500 CHARACTERS OF BEING FILLED. THE INPUT WILL TERMINATE WITH THE NEXT CR-LF COMBINATION.

UNDER THE SINGLE JOB MONITOR, AN 8K SYSTEM MAY HOLD ABOUT 6,000 CHARACTERS IN ITS BUFFERS. THIS IS EXPANDED BY 8,000 CHARACTERS FOR EACH ADDITIONAL 4K OF MEMORY.

SYSTEM DEVICE STORAGE ALLOCATION

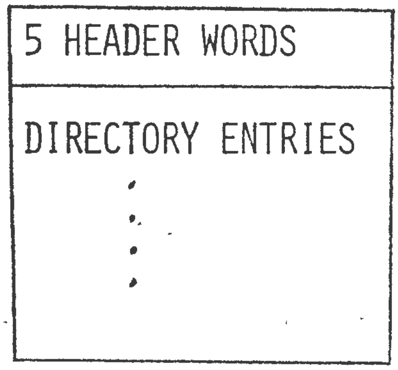


DIRECTORY SEGMENTS

FORMAT

2 LOGICAL BLOCKS

512 WORDS



INITIALIZE/SEGMENTS:n - SPECIFY NUMBER OF DIRECTORY SEGMENTS. LIMITS 1-31

DEFAULT NUMBER OF DIRECTORY SEGMENTS:

RK	16
DT	4
RF	4
DS	4
DP	31
DX	4 ← each segment controls up to 72 files.
DM	31

DIRECTORY HEADER FORMAT

WORD	CONTENTS
1	THE NUMBER OF SEGMENTS AVAILABLE FOR ENTRIES; THIS NUMBER IS SPECIFIED WITH THE COMMAND INITIALIZE WHEN THE DEVICE IS ZEROED AND MUST BE IN THE RANGE $1 \leq N \leq 31_{10}$.
2	SEGMENT NUMBER OF THE NEXT LOGICAL DIRECTORY SEGMENT; THE DIRECTORY MAY BE IN CERTAIN CASES, A LINKED LIST. THIS WORD IS THE LINK WORD BETWEEN LOGICALLY CONTIGUOUS SEGMENTS: IF EQUAL TO 0, THERE ARE NO OTHER SEGMENTS IN THE LIST.
3	THE HIGHEST SEGMENT CURRENTLY OPEN; EACH TIME A NEW SEGMENT IS CREATED, THIS NUMBER IS INCREMENTED. THIS WORD IS ONLY USED OR UPDATED IN THE FIRST SEGMENT.
4	THE NUMBER OF EXTRA BYTES PER DIRECTORY ENTRY THIS NUMBER CAN BE SPECIFIED WHEN THE DEVICE IS ZEROED WITH THE COMMAND INITIALIZE. CURRENTLY, RT-11 DOES NOT ALLOW DIRECT MANIPULATION OF INFORMATION IN THE EXTRA BYTES.
5	BLOCK NUMBER WHERE FILES IN THIS SEGMENT BEGIN

DIRECTORY ENTRY FORMAT

WORD 1	STATUS WORD		PERM. Tentative empty when you delete a file the statu is empty.
2	NAME (CHARS 1-3)		IN RAD 50
3	NAME (CHARS 4-6)		
4	TYPE		
5	TOTAL FILE LENGTH		in blocks
6	Backgnd=0 JOB # foregnd = 2	CH # 0-15	
7	CREATION DATE		
8	EXTRA WORDS		OPTIONAL
•	•		
•	•		
•	•		

DIRECTORY ENTRY STATUS WORD

THE STATUS WORD IS BROKEN DOWN INTO TWO BYTES OF DATA:

LOW ORDER:	RESERVED FOR FUTURE USE
HIGH ORDER:	INDICATES THE TYPE OF ENTRY. RT-11 CURRENTLY RECOGNIZES THE FILE TYPES LISTED IN TABLE 4.

VALUE	FILE TYPE
1	TENTATIVE FILE; I.E. ONE THAT HAS BEEN CREATED BUT NOT CLOSED. FILES OF THIS TYPE ARE DELETED IF NOT EVENTUALLY CLOSED AND ARE LISTED BY THE COMMAND DIRECTORY AS UNUSED FILES.
2	AN EMPTY FILE: THE NAME, EXTENSION AND DATE FIELDS ARE NOT USED. THE COMMAND DIRECTORY LISTS AN EMPTY FILE AS UNUSED FOLLOWED BY THE LENGTH OF THE UNUSED AREA.
4	A PERMANENT ENTRY; A TENTATIVE FILE THAT HAS BEEN CLOSED IS A PERMANENT FILE. THE NAME OF A PERMANENT FILE IS UNIQUE; THERE CAN BE ONLY ONE FILE WITH A GIVEN NAME AND EXTENSION. IF ANOTHER EXISTS BEFORE THE FILE IS CLOSED, IT IS DELETED BY THE MONITOR.
10	END-OF-SEGMENT MARKER; RT-11 USES THIS TO DETERMINE WHEN THE END OF THE DIRECTORY SEGMENT HAS BEEN REACHED DURING A DIRECTORY SEARCH. THE DIRECTORY ENTRY FOR AN END-OF-SEGMENT MARKER IS ONLY ONE WORD.

TO LOCATE FILE ON RT-11 VOLUME

E.G. ABCDEF.XYZ

1. SEARCH DIRECTORY SEGMENTS FOR FILE SPECIFIER (E.G. ABCDEF.XYZ)

2. WORD 5 OF THAT DIRECTORY SEGMENT HEADER GIVES FIRST BLOCK NUMBER OF FIRST FILE IN SEGMENT (E.G. 2014)

3. WORD 5 OF EACH PRECEDING DIRECTORY ENTRY IN THAT SEGMENT WILL YIELD SIZE OF EACH PRECEDING FILE. THE SUM OF THESE WORD 5 VALUES GIVES THE NUMBER OF BLOCKS PRIOR TO OUR FILE. (E.G. 51)

4. SUM OF #2 AND #3 GIVES STARTING BLOCK NUMBER OF OUR FILE (E.G. 2014 & 51 = 2065)

INDIRECT COMMAND FILE EXAMPLE

FIRST.COM

```
! COMMAND FILE
DATE
TIME
INITIALIZE/VOLUMEID DT0:

^C
@ SECOND     !NESTED CMD FILE
PRINT DATA.DAT
DIRECTORY JOB.*
DELETE/NOQUERY JOB.*
DIRECTORY JOB.*
! END OF FILE
```

SECOND.COM

```
MACRO/LIST JOB
LINK/MAP JOB
RUN JOB
```

ODT SESSION

```
.MAIN.   MACRO V03.01   00:24:09 PAGE 1


      1                                          .MCALL  .PRINT,.EXIT
      2 000000  012700  000050'          START:  MOV     #TABLE,R0
      3 000004  012701  000067'                  MOV     #OUTTBL,R1
      4 000010  012702  177773                   MOV     #-5,R2
      5 000014  112767  000026           MOVE:   MOVB    (R0)+,TEMP
      6 000020  062767  000060  000020           ADD     #60,TEMP
      7 000026  116721  000014                   MOVB    TEMP,(R1)+
      8 000032  005202                           INC     R2
      9 000034  001367                           BNE     MOVE
     10 000036                                   .PRINT  #MSG
     11 000044                                   .EXIT
     12 000046  000000                   TEMP:   .WORD
     13 000050     002     003     004   TABLE:  .BYTE   2,3,4,5,6
        000053     005     006
     14 000055     116     125     115   MSG:    .ASCII  /NUMBERS   /
        000060     102     105     122
        000063     123     040     040
        000066     040
     15 000067                           OUTTBL: .BLKB   5
     16 000074     000                           .BYTE   0
     17                                          .EVEN
     18         000000'                          .END START
```

```
.LINK/MAP:TT: PROG,ODT/TRANSFER
Transfer symbol? O.ODT
RT-11 LINK   V05.02       Load Map
PROG   .SAV        Title:  .MAIN.   Ident:

Section   Addr    Size     Global   Value    Global   Value    Global   Value

. ABS.   000000   001000    (RW,I,GBL,ABS,OVR)
         001000   006176    (RW,I,LCL,REL,CON)
                           O.ODT    001330

Transfer address = 001330, High limit = 007176 =  1855. words

.
```

ODT SESSION

```
.RUN PROG

 ODT  V01.06
*1010/012702
*1000;R
*0,10/012702
0,000012 /177773 ^
0,000010 /012702 12705
0,000012 /177773 ^
0,000010 /012705 12702
*$6/001000
*0,50\002 =?
0,000051 \003 =? ^
0,000050 \002 =?
*0,55;5ANUMBE

*0,34/001367 X= R9
*0,36;0B
*0,14;1B
*0,0;G
B1;0,000014
*;P
B1;0,000014
*0,67\062 =2
0,000070 \000 =
*2;P
B1;0,000014
*0,67\062 =2
0,000070 \063 =3
0,000071 \064 =4
0,000072 \000 =
*$B/001036
0,000244 /001014
0,000246 /007174
*;1B
*;P
B0;0,000036
*$B/001036
0,000244 /007174
0,000246 /007174
*$R/001000
0,000224 /177777
0,000226 /177777
*;P
NUMBERS   23456

.
```

FDT SESSION

```
FORTRAN IV      V02.04                                                 PAGE 001

0001          DIMENSION K(10),A(10)
0002          DO 100 I=1,10
0003          K(I)=I**2
0004          A(I)=FLOAT(K(I))/17.
0005          J=J+K(I)
0006  100     CONTINUE
0007          WRITE (5,1) J
0008  1       FORMAT (1X,I10)
0009          END
.MAIN.
```

```
FORTRAN IV      Storage Map for Program Unit .MAIN.

Local Variables, .PSECT $DATA, Size = 000104 (    34. words)

Name    Type   Offset       Name    Type   Offset       Name    Type   Offset
I       I*2    000100       J       I*2    000102

Local and COMMON Arrays:

Name      Type     Section Offset  ------Size----- Dimensions
A       R*4      $DATA   000024  000050 (   20.) (10)
K       I*2      $DATA   000000  000024 (   10.) (10)

Subroutines, Functions, Statement and Processor-Defined Functions:

Name    Type    Name    Type    Name    Type    Name    Type    Name    Type
FLOAT    R*4

.



.LINK/DEBUG:FDT PROG

.
```

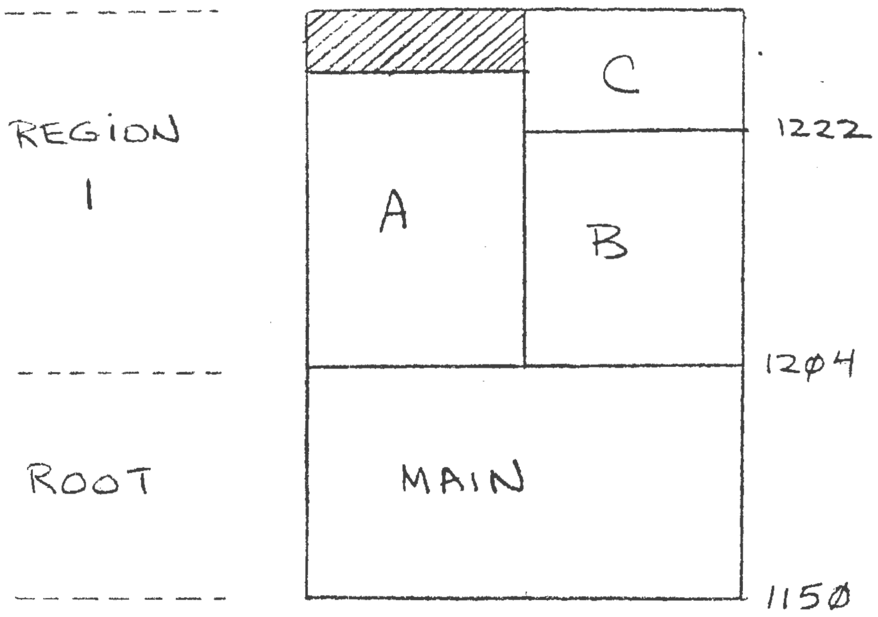
FDT SESSION

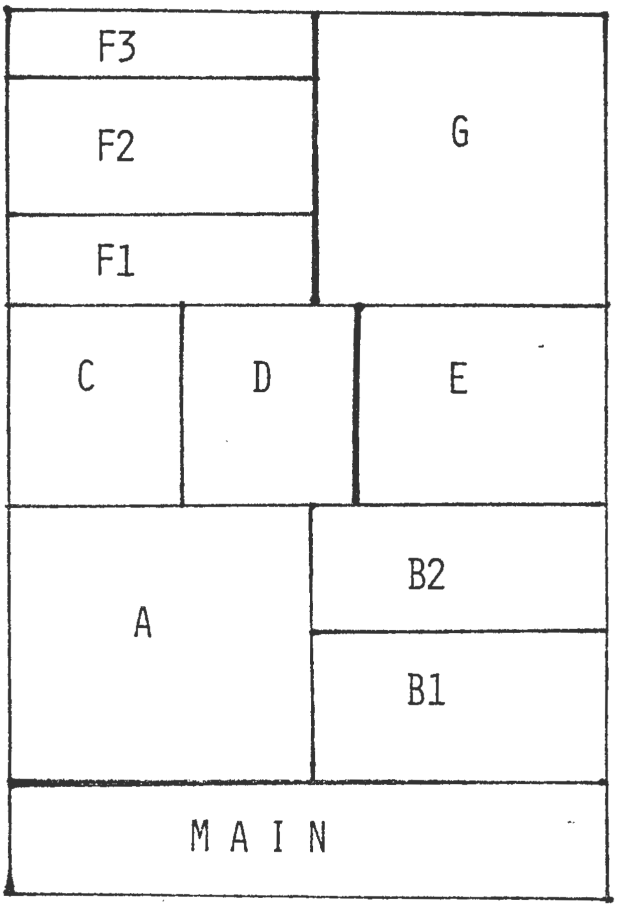
```
.RUN PROG
FDT     V02-02

FDT PAUSE AT ISN 2 IN .MAIN.
!NAME I,100
!NAME J,102
!DIMENSION K(10),0
!DIMENSION A(10),24
!PAUSE .MAIN.,6
!START

FDT PAUSE AT ISN 6 IN .MAIN.
!TYPE I
1
!TYPE J,K(1),K(2)
1,1,0
!CONTINUE

FDT PAUSE AT ISN 6 IN .MAIN.
!TYPE I,J,K(2)
2,5,4
!CONTINUE 3

FDT PAUSE AT ISN 6 IN .MAIN.
!TYPE I,J
5,55
!RESET .MAIN.,6
!PAUSE .MAIN.,7
!CONTINUE

FDT PAUSE AT ISN 7 IN .MAIN.
!TYPE I,J,K(10)
11,385,100
!CON
       385
STOP --

.
```

RT-11 OVERLAYS

ROOT

```
.LINK/PROMPT/MAP:TT: MAIN
*A/O:1
*B,C/O:1
*//
RT-11 LINK  V05.02        Load Map
MAIN  .SAV        Title:  .MAIN.  Ident:

Section  Addr    Size     Global  Value    Global  Value    Global  Value

. ABS.  000000  001150    (RW,I,GBL,ABS,OVR)
        001150  000032    (RW,I,LCL,REL,CON)
 Segment size = 001202 =    321. words

 Overlay region 000001  Segment 000001
        001204  000016    (RW,I,LCL,REL,CON)
                         $A     @ 001204
 Segment size = 000016 =      7. words

 Overlay region 000001  Segment 000002
        001204  000034    (RW,I,LCL,REL,CON)
                         $B     @ 001204   $C     @ 001222
 Segment size = 000034 =     14. words

Transfer address = 001150, High limit = 001240 =    336. words
```




F3
F2
F1
G
C
D
E
A
B2
B1
MAIN

BASIC SESSION

```
.BASIC
BASIC-11/RT-11 V02-03
OPTIONAL FUNCTIONS (ALL, NONE, OR INDIVIDUAL)? A

READY
10 PRINT "THIS IS A BASIC PROGRAM"
20 FOR I=1 TO 5
30 INPUT J
40 T=T+M
50 NEZT I
60 PRINT "THE TOTAL IS  ";T
999 END
40 T=T+J
SUB 50 @NEZ@NEX
50 NEXTI

READY
45 LET X=1
43 IF J=X THEN GOTO 60
45
DEL 41-44

READY
LIST


NONAME                    00:19:46

10 PRINT "THIS IS A BASIC PROGRAM"
20 FOR I=1 TO 5
30 INPUT J
40 T=T+J
50 NEXT I
60 PRINT "THE TOTAL IS  ";T
999 END

READY
RUN


NONAME                    00:20:01

THIS IS A BASIC PROGRAM
? 1
? 2
? 3
? 4
? 5
THE TOTAL IS   15

READY
SAVE PROG

READY
```

BASIC SESSION

```
NEW
NEW FILE NAME--FRAMUS

READY
10 PRINT 'THIS IS A NEW PROGRAM - FRAMUS'
RUN


FRAMUS                    00:21:23

THIS IS A NEW PROGRAM - FRAMUS

READY
OLD PROG

READY
RUN


PROG                      00:21:45

THIS IS A BASIC PROGRAM
? 4
? 3
? 5
? 2
? 6
THE TOTAL IS    20

READY
SCR

READY
RUN


NONAME                    00:22:04


READY
BYE

+
```

FOREGROUND JOB

```
.SHOW
TT (RESIDENT)
RK (RESIDENT)
  RKO  = SY , DK
DM
DS
RF
DP
DX
DT  (LOADED)
  DTO (F)
MT  (LOADED=B)
MM
CT
LP
BA
 FREE

USR SWAP
```

ODT IN FOREGROUND - SUG P.16-17

LINK PROG IN FOREGROUND

LINK ODT IN BACKGROUND

```
.FRUN PROG/P
 LOADED AT 13456
.RUN ODT
 ODT V01.1
*13456;R
*$F/000000
*0,50;0B
*0;G
.RESUME
```

MONITOR SERVICES AVAILABLE THROUGH SYSGEN

MULTI-TERMINAL SUPPORT (FB,XM)

PROCESSING OF ESCAPE SEQUENCES

BY TERMINAL HANDLER

ERROR LOGGING

IDLE LOOP LIGHT PATTERN (FB,XM)

TIMER SUPPORT (SJ)

ERROR MESSAGES ON SYSTEM I/O

ERRORS (SJ)

XM MONITOR

```
.RUN SYSGEN

RT-11 SYSTEM GENERATION PROGRAM V02.11


DO YOU WANT THE EXPANDED FORM OF DIALOGUE [Y/N] (Y)? N

DO YOU WANT THE BASE-LINE SINGLE-JOB MONITOR [Y/N] (N)? N

DO YOU WANT THE SINGLE-JOB (SJ) MONITOR [Y/N] (Y)? Y

DO YOU WANT THE FOREGROUND/BACKGROUND (FB) MONITOR [Y/N] (Y)?

DO YOU WANT THE EXTENDED MEMORY (XM) MONITOR [Y/N] (N)?

DO YOU WANT TIMER SUPPORT IN THE SJ MONITOR [Y/N] (N)?

DO YOU WANT DEVICE TIME-OUT SUPPORT [Y/N] (N)?

DO YOU WANT AN ERROR MESSAGE ON SYSTEM I/O ERRORS [Y/N] (Y)

DO YOU WANT IDLE LOOP LIGHT PATTERN [Y/N] (N)? Y

DO YOU WANT SUPPORT OF MORE THAN ONE TERMINAL [Y/N[ (N)? Y

DO YOU WANT THE OPTIONAL 50 HZ CLOCK SUPPORT [Y/N] (N)?

DO YOU WANT TO USE THE KW11-P CLOCK AS THE SYSTEM CLOCK [Y/N] (N)?

DO YOU WANT THE STARTUP INDIRECT FILE [Y/N] (Y)? Y

DO YOU WANT MEMORY PARITY SUPPORT [Y/N] (N)? Y

DO YOU WANT POWER FAILURE MESSAGES [Y/N] (Y)?

DO YOU WANT BATCH SUPPORT [Y/N] (N)? Y

DO YOU WANT ESCAPE SEQUENCE PROCESSING [Y/N] (N)?

DO YOU WANT ERROR LOGGING [Y/N] (N)?
```

```
WHAT IS THE NAME OF THE SYSTEM DEVICE [DD] (RK)? RK

DO YOU WANT RJS03 OR RJS04 DISK SUPPORT [Y/N (N)?

DO YOU WANT RF11 FIXED-HEAD DISK SUPPORT [Y/N] (N)?

DO YOU WANT RP11/RPR02 OR RP11/RP03 DISK SUPPORT [Y/N] (N)?

DO YOU WANT RX01/RX11 FLEXIBLE DISKETTE SUPPORT [Y/N] (N)? Y

DO YOU WANT SUPPORT FOR A SECOND RX11 CONTROLLER [Y/N] (N)?

WHAT IS THE CSR ADDRESS FOR THE FIRST RX11 CONTROLLER [0] (177170)?

WHAT IS THE VECTOR ADDRESS FOR THE FIRST RX11 CONTROLLER [0] (264)?

DO YOU WANT RX02 DISKETTE SUPPORT [Y/N] (N)?

DO YOU WANT RL01 DISK SUPPORT [Y/N] (N)?

DO YOU WANT TC11 DECTAPE SUPPORT [Y/N] (N)? Y

DO YOU WANT RK06/RK07 DISK SUPPORT [Y/N] (N)?

DO YOU WANT MAGNETIC TAPE SUPPORT [Y/N] (N)?

DO YOU WANT TA11 CASSETTE SUPPORT [Y/N] (N)? Y

DO YOU WANT VS60 SUPPORT [Y/N] (N)?

VT11 HARDWARE IS ASSUMED.

WHAT IS THE CSR ADDRESS FOR THE VT11/VS60 [0] (172000)?

WHAT IS THE VECTOR ADDRESS FOR THE VT11/VS60 [0] (320)?

DO YOU WANT LINE PRINTER SUPPORT [Y/N] (Y)? Y

DOES YOUR PRINTER HAVE A NON-STANDARD VECTOR OR CSR ADDRESS [Y/N] (N)?
```

```
DO YOU WANT PC11 HIGH-SPEED PAPER TAPE READER/PUNCH SUPPORT [ /N] (N)? Y

DO YOU WANT CARD READER SUPPORT [Y/N] (N)?

DO YOU WANT THE NULL HANDLER [Y/N] (Y)? Y

HOW MANY EXTRA DEVICE SLOTS DO YOU WANT [D]  (0)?

DO YOU WANT TO RETAIN THE SYSTEM OBJ'S [Y/N] (N)? Y

HOW MANY LOCAL DL11'S, INCLUDING THE CONSOLE [D] (1)? 5

HOW MANY REMOTE DL11 LINES [D] (0)?

WHAT IS THE CSR ADDRESS FOR THE FIRST (CONSOLE) DL11 [0] (177560)?

WHAT IS THE VECTOR ADDRESS FOR THE FIRST (CONSOLE) DL11 [0] (60)?

WHAT IS THE CSR ADDRESS FOR THE NEXT (LOCAL) DL11 [0] (176500)?

WHAT IS THE VECTOR ADDRESS FOR THE NEXT (LOCAL) DL11 [0] (300)? 310

WHAT IS THE CSR ADDRESS FOR THE NEXT (LOCAL) (DL11 [0] (176510)?
WHAT IS THE VECTOR ADDRESS FOR THE NEXT (LOCAL DL11 [0] (310)? 320

WHAT IS THE CSR ADDRESS FOR THE NEXT (LOCAL) DL11 [0] (176520)?

WHAT IS THE VECTOR ADDRESS FOR THE NEXT (LOCAL) DL11 [0] (320)? 330

WHAT IS THE CSR ADDRESS FOR THE NEXT (LOCAL) DL11 [0] (176530)?

WHAT IS THE VECTOR ADDRESS FOR THE NEXT (LOCAL) DL11 [0] (330)? 340
```

for V3.0 will ask for NEXT N 2

(after all spec. are taken care of

```
DO YOU WANT DZ11 OR DZV11 MULTIPLEXOR SUPPORT [Y/N] (N)?

SIZE OF THE OUTPUT BUFFERS IN CHARACTERS [D] (40)?

SIZE OF THE INPUT BUFFERS IN CHARACTERS [D] (82)?

DO YOU WANT ASYNCHRONOUS TERMINAL STATUS [Y/N] (Y)?
DO YOU WANT MULTI-TERMINAL TIME-OUT SUPPORT [Y/N] (Y)?

WHAT IS THE PHYSICAL NAME AND UNIT OF THE SOURCE INPUT
     DEVICE [DDU] (RK1)? RK0

WHAT IS THE PHYSICAL NAME AND UNIT OF THE BINARY OUTPUT DEVICE
     [DDU] (RK0)? RK1

WHAT IS THE PHYSICAL NAME AND UNIT OF THE MAP OUTPUT DEVICE
     [DDU] (TT)?

END OF SYSGEN PROGRAM -- DONT'T FORGET TO COPY DK:SYCND.MAC AND
     DK:SYSTBL.MAC TO THE SOURCE DISK BEFORE EXECUTING THE
     COMMAND FILE(S).  REMEMBER THAT SWAP.SYS MUST BE ON ALL
     BOOTABLE VOLUMES.


@SYSBLD

.

.@MONBLD

.

.ASS RK0 SRC

.ASS RK1 BIN

.ASS TT MAP

. MACRO/OBJ:BIN:KMSJ SRC:(SJ+SYCND+EDTGBL+KMON+KMOVLY)
ERRORS DETECTED:  0

.MACRO/OBJ:BIN:RMSJ SRC:(SJ+SYCND+EDTGBL+USR+RMONSJ)
ERRORS DETECTED:  0

.@DEVBLD

.

.ASSRK0 SRC

.ASS RK1 BIN
.MACRO/OBJ:BIN:TT SRC:(SYNCND+TT)
ERRORS DETECTED:  0

.LINK/EXE:BIN:TT.SYG BIN:TT
```

```
.TYPE DEMO.BAT



$JOB
$MESSAGE        THIS IS A SAMPLE BATCH STREAM
$MESSAGE        NOW CREATE A MACRO PROGRAM FILE
$CREATE/LIST    EXAMPL.MAC
        .LIST   TTM
        .NLIST  BEX
        .TITLE  BATCH
        .MCALL .PRINT,.EXIT
START:  .PRINT  #MESSAG
        .EXIT
MESSAG: .ASCIZ  /SAMPLE MACRO PROGRAM FOR BATCH/
        .END START
$EOD
$MESSAGE        NOW ASSEMBLE AND LIST THE MACRO PROGRAM
$MACRO EXAMPL EXAMPL/OBJECT EXAMPL/LIST
$LINK EXAMPL EXAMPL/EXECUTE
$PRINT/DELETE EXAMPL.LST
$MESSAGE        NOW RUN THE MACRO PROGRAM
$RUN EXAMPL
$DELETE EXAMPL.*
$MESSAGE        NOW AN EXAMPLE OF RT-11 MODE
$RT11
        TTYIO
        LET N="0
.TYPE D.TXT
LOOP:
.MACRO PROG'N'
.LINK PROG'N'
        %N
        IF(N-"2)-LOOP,-LOOP,END
END:
$MESSAGE        END OF BATCH STREAM EXAMPLE
$EOJ
```

BATCH SESSION

```
.LOA BA:

.ASSIGN TT: LST

.ASSIGN RK: LOG

.RUN BATCH
*,DEMO[10]=DEMO
 THIS IS A SAMPLE BATCH STREAM
 NOW CREATE A MACRO PROGRAM FILE
 NOW ASSEMBLE AND LIST THE MACRO PROGRAM

BATCH   MACRO V03.01  00:36:05 PAGE 1


      1                         .LIST   TTM
      2                         .NLIST  BEX
      3                         .TITLE  BATCH
      4                         .MCALL .PRINT,.EXIT
      5 000000          START:  .PRINT  #MESSAG
      6 000006                  .EXIT
      7 000010     123  MESSAG: .ASCIZ  /SAMPLE MACRO PROGRAM FOR BATCH/
      8         000000'         .END START






BATCH   MACRO V03.01  00:36:05 PAGE 1-1
SYMBOL TABLE

MESSAG  000010R         START   000000R

. ABS.  000000     000
        000047     001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  503 WORDS  ( 2 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  54 PAGES
EXAMPL,EXAMPL=EXAMPL

 NOW RUN THE MACRO PROGRAM
 NOW AN EXAMPLE OF RT-11 MODE
AN NOW AN ASSEMBLY AND LINK
 END OF BATCH STREAM EXAMPLE

END BATCH

.
```

```
 TYPE DEMO.LOG



$JOB

$MESSAGE          THIS IS A SAMPLE BATCH STREAM

$MESSAGE          NOW CREATE A MACRO PROGRAM FILE

$CREATE/LIST      EXAMPL.MAC

        .LIST     TTM
        .NLIST    BEX
        .TITLE    BATCH
        .MCALL    .PRINT,.EXIT
START:  .PRINT    #MESSAG
        .EXIT
MESSAG: .ASCIZ    /SAMPLE MACRO PROGRAM FOR BATCH/
        .END START


$EOD

$MESSAGE          NOW ASSEMBLE AND LIST THE MACRO PROGRAM

$MACRO EXAMPL EXAMPL/OBJECT EXAMPL/LIST

ERRORS DETECTED:   0

$LINK EXAMPL EXAMPL/EXECUTE


$PRINT/DELETE EXAMPL.LST



$MESSAGE          NOW RUN THE MACRO PROGRAM

$RUN EXAMPL

SAMPLE MACRO PROGRAM FOR BATCH

$DELETE EXAMPL.*


$MESSAGE          NOW AN EXAMPLE OF RT-11 MODE

$RT11
        TTYIO
        LET N="0

TYPE D.TXT
```

```
LOOP:

MACRO PROGO
ERRORS DETECTED:   0


LINK PROGO

        %N
        IF(N-'2)-LOOP,-LOOP,END
LOOP:

MACRO PROG1
ERRORS DETECTED:   0


LINK PROG1

        %N
        IF(N-'2)-LOOP,-LOOP,END
LOOP:

MACRO PROG2
ERRORS DETECTED:   0


LINK PROG2

        %N
        IF(N-'2)-LOOP,-LOOP,END
END:

$MESSAGE        END OF BATCH STREAM EXAMPLE

$EOJ
```

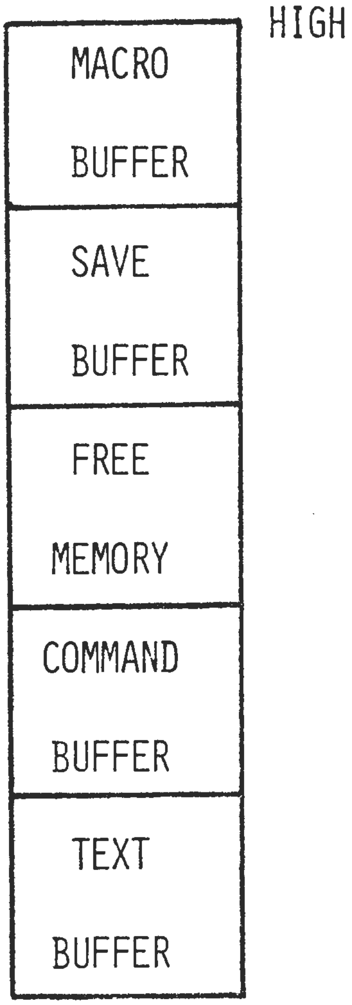
ADDITIONAL EDITOR COMMANDS

nS- COPIES n LINES BEGINNING AT POINTER INTO SAVE BUFFER

U- INSERTS THE CONTENTS OF THE SAVE BUFFER INTO THE TEXT BUFFER AT THE POINTER.

M/COMMAND STRING/ - STORES THE COMMAND STRING INTO THE MACRO BUFFER.

nEM- EXECUTES COMMAND STRING IN MACRO BUFFER n TIMES.



RT-11 EDITOR REFERENCE

$ - ESCAPE (ALT MODE)

B - MOVE POINTER TO BEGINNING OF TEXT BUFFER.

V - PRINT LINE TO WHICH POINTER POINTS.

GSTRING$ - LOCATE STRING IN TEXT BUFFER.

ISTRING$ - INSERT STRING IN TEXT BUFFER AT POINTER.

nCSTRING$ - CHANGE n CHARACTERS FROM POINTER INTO STRING.

EX - WRITE OUTPUT FILE AND TERMINATE EDITOR.

R - READ NEXT PAGE INTO TEXT BUFFER. NOTHING WRITTEN TO OUTPUT FILE.

N - READ NEXT PAGE INTO TEXT BUFFER. WRITE PREVIOUS PAGE TO OUTPUT FILE.

RT-11 EDITOR REFERENCE

LINE ORIENTED

CHARACTER ORIENTED

COMMAND	LIST	ADVANCE	KILL		JUMP	DELETE
n CHARACTERS OR LINES	nL	nA	nK		nJ	nD
-n CHARACTERS OR LINES	-nL	-nA	-nK		-nJ	-nD
0 BEGINNING OF LINE TO POINTER	0L	0A	0K		0J	0D
/ POINTER TO END OF TEXT BUFFER	/L	/A	/K		/J	/D
EFFECT ON POINTER	STATIC	DYNAMIC	STATIC (TEXT MOVES)		DYNAMIC	STATIC (TEXT MOVES)

RT-11 ODT REFERENCE

N;R - SETS NEXT RELOCATION REGISTER TO N

/ (SLASH) - OPEN WORD LOCATION AND DISPLAY CONTENTS

\ (BACKSLASH) - OPEN BYTE LOCATION AND DISPLAY CONTENTS

LINE FEED - CLOSE CURRENT LOCATION, OPEN AND DISPLAY NEXT LOCATION

UP-ARROW (↑) - CLOSE CURRENT LOCATION, OPEN AND DISPLAY PREVIOUS LOCATION

N;KA - DISPLAY ASCII VALUES FOR K BYTES STARTING AT LOCATION N.

X - DISPLAY RA050 VALUE FOR CURRENT WORD LOCATION

N;KB - SET BREAKPOINT K AT LOCATION N

;KB - REMOVE BREAKPOINT K

K;P - PROCEED, STOP AT K BREAKPOINT ENCOUNTERED

$B - DISPLAY BREAKPOINTS

$R - DISPLAY RELOCATION REGISTERS

CTRL-C - EXIT PROGRAM, RETURN TO MONITOR

RT-11 ODT REFERENCE (CONT'D).

SINGLE INSTRUCTION MODE

;2S - ENTER SINGLE INSTRUCTION MODE
2 HAS NO SPECIAL MEANING

N;P - EXECUTE N INSTRUCTIONS IN PROGRAM

;S - LEAVE SINGLE INSTRUCTION NODE

RT-11 FDT REFERENCE

NAME I,20 - ASSOCIATES INTEGER VARIABLE I WITH OFFSET 20

DIMENSION K(5),100 - ASSOCIATE 5 ELEMENT INTEGER ARRAY K WITH OFFSET 100

NAME VAR/E,36,7 - ASSOCIATES REAL VARIABLE VAR WITH OFFSET 36

PAUSE .MAIN.,6 - SETS A PAUSE AT ISN 6 IN MAIN PROGRAM

RESET FACTOR,5 - DELETES A PAUSE AT ISN 5 IN SUBROUTINE FACTOR

START - EXECUTES YOUR PROGRAM

CONTINUE 3 - CONTINUE PROGRAM EXECUTION, PASS 3 PAUSES, DEFAULT 1

STEP 5 - EXECUTE 5 STATEMENTS

TYPE K(3) - TYPE VALUE OF K(3) ON TERMINAL

ACCEPT I=4 - SETS VALUE OF I TO 4

RT-11 BASIC-11 REFERENCE

DEL 30-40	- ERASE PROGRAM LINES 30-40 INCLUSIVE
LIST	- LIST ENTIRE PROGRAM AND PRINT HEADER
SUB 20 @ABC@DE	- REPLACE ABC IN LINE 20 WITH DE
RUN	- EXECUTE YOUR BASIC PROGRAM
CTRL/C,CTRL/C	- TERMINATE YOUR BASIC PROGRAM, RETURN TO INTERPRETER
BYE	- EXIT BASIC INTERPRETER RETURN TO RT-11 MONITOR
SAVE RK0:SAM	- STORE BASIC PROGRAM ON DISK AS SAM.BAS
OLD	- BRING INTO MEMORY A BASIC PROGRAM FROM DISK
NEW	- ERASES MEMORY AND NAMES NEW PROGRAM
SCR	- ERASES MEMORY

MONITOR COMMAND LAB

1. BOOT YOUR DISK. SET TIME AND DATE.
2. DISPLAY ALL DEVICES KNOWN TO YOUR MONITOR.
3. LOAD THE HANDLERS FOR DECTAPE AND MAG TAPE.
4. MAKE LOGICAL ASSIGNMENTS OF FL FOR THE FLOPPY DISK. AND TAP FOR THE MAG TAPE.
5. REPEAT STEP #2.
6. DELETE ALL LOGICAL ASSIGNMENTS.
7. UNLOAD THE HANDLERS FOR DECTAPE AND MAG TAPE.
8. BOOT THE FOREGROUND-BACKGROUND MONITOR FROM YOUR DISK.
9. TYPE ABCDEF AND 3 DELETES. TYPE CTRL/U.
10. DESIGNATE THE CONSOLE AS A SCOPE(SET).
11. REPEAT STEP #9.
12. SET TERMINAL WIDTH TO 30. VERIFY BY TYPING MORE THAN 30 CHARACTERS.
13. SET NOCRLF OPTION INTO TERMINAL AND REPEAT STEP #12.
14. BOOT THE SINGLE JOB MONITOR. SET TIME AND DATE.
15. GET A DIRECTORY LISTING OF YOUR DISK.
16. MAKE COPIES OF 3 .SAV FILES WITH FILE TYPE .DEC.
17. MAKE A COPY OF BA.SYS CALLED BATCH. DEC.
18. MAKE A COPY OF ODT.OBJ WITH FILE TYPE .DEC WHICH WILL HAVE TODAY'S DATE.
19. CONCATENATE 2 .SAV FILES TOGETHER INTO A FILE CALLED ABC.DEC.

20. GET A DIRECTORY LISTING OF:

    A) ALL .DEC FILES
    B) ALL .DEC FILES-NAME AND TYPE ONLY
    C) ALL FILES WITH TODAY'S DATE.
    D) ALL .DEC FILES IN ALPHABETICAL ORDER.
    E) ALL .DEC FILES IN SINGLE COLUMN FORMAT.
    F) THE ENTIRE DISK BEGINNING WITH THE 20TH FILE. * (2 cmds)
    G) THE ENTIRE DISK EXCEPT .SYS AND .SAV FILES.
    H) ALL UNUSED SPACE ON YOUR DISK.

* NOT ONE COMMAND

21. GENERATE A FILE CONTAINING A DIRECTORY LISTING OF ALL .DEC FILES IN REVERSE ORDER OF SIZE.

22. PRINT THE ABOVE FILE ON YOUR TERMINAL.

23. DISPLAY THE VOLUME IDENTIFICATION FOR YOUR DISK.

24. FIND THE STARTING BLOCK NUMBER OF ABC. DEC.

25. CHANGE THE NAME OF ABC.DEC TO XYZ.DEC.

26. DETERMINE IF THERE ARE ANY BAD BLOCKS ON YOUR DISK (DO NOT USE INITIALIZE). IN WHICH FILE?

THE FOLLOWING STEPS REQUIRE A SECOND MASS STORAGE DEVICE (DECTAPE,RK05, ETC).

27. INITIALIZE THE SECONDARY DEVICE TO HAVE VOLUME-ID "PRACTICE".

28. COPY ALL .DEC FILES TO THAT SECONDARY DEVICE.

29. DELETE BATCH .DEC AND XYZ.DEC FROM THE SECONDARY DEVICE.

30. COPY 2 .DEC FILES FROM YOUR DISK TO THE SECONDARY DEVICE, ONE WITH THE OPTION "REPLACE" AND THE OTHER WITH THE OPTION "PREDELETE".

31. GET A DIRECTORY LISTING OF THE SECONDARY DEVICE.

32. DELETE ALL .DEC FILES EXCEPT ONE FROM YOUR DISK.

. SET TOMORROW'S DATE INTO THE SYSTEM.

34. COPY ALL .DEC FILES FROM THE SECONDARY DEVICE TO YOUR DISK USING THE OPTIONS "SETDATE" AND "NOREPLACE".

35. GET A DIRECTORY LISTING OF THE SECONDARY DEVICE OF ALL UNUSED SPACE.

36. SQUEEZE THE SECONDARY DEVICE.

37. REPEAT STEP #36.

38. DELETE ALL .DEC FILES FROM YOUR DISK.

39. MAKE YOUR SECONDARY DEVICE INTO A BOOTABLE RT-11 DEVICE

    A. COPY THE FOLLOWING FILES TO THE SECONDARY DEVICE
    SINGLE JOB MONITOR FILE (E.G. DTMNSJ.SYS)

    RK.SYS
    SWAP.SYS
    TT.SYS
    PIP.SAV
    DUP.SAV
    DIR.SAV
    DUMP.SAV

    B. WRITE THE BOOTSTRAP TO YOUR SECONDARY DEVICE (COPY/BOOT)

    C. BOOT THE SECONDARY DEVICE.

40. COPY SAMPLE.DMP FROM YOUR DISK TO YOUR SECONDARY DEVICE.

41. DUMP THE CONTENTS OF SAMPLE.DMP IN WORD, BYTE AND RAD50 FORMATS.

PROGRAM DEVELOPMENT LAB

1. BOOT YOUR DISK. BOOT THE FOREGROUND - BACKGROUND MONITOR. SET THE DATE AND TIME.

2. PROGRAMS HELLO.MAC AND HELLO.FOR HAVE BEEN TYPED ONTO YOUR DISK.

3. GET A DIRECTORY LISTING OF ALL FILES WITH NAME HELLO.

4. ASSEMBLE/COMPILE HELLO. GET LISTING ON YOUR TERMINAL.

5. REPEAT STEP #3.

6. LINK HELLO. GET MAP ON YOUR TERMINAL.

7. REPEAT STEP #3

8. RUN HELLO. TYPE YOUR NAME AS REQUESTED.

9. ASSEMBLE/COMPILE HELLO AGAIN. GET LISTING IN A FILE HELLO.LST.

10. REPEAT STEP #3. WHAT HAPPENED TO OLD HELLO. OBJ?

11. GET COPY OF LISTING FILE ON YOUR TERMINAL.

12. MODIFY YOUR SOURCE FILE AS FOLLOWS:

```
HELLO.MAC
    CHANGE       MOV #5,R2
      TO         MOV #7,R2

HELLO.FOR
    CHANGE       DO 100 I=1,5
      TO         DO 100 I=1,7
```

13. REPEAT STEP #3. WHAT IS HELLO.BAK? GET COPY ON YOUR TERMINAL.

14. USE THE EXECUTE COMMAND TO RUN THE MODIFIED HELLO. GET LISTING AND MAP ON YOUR TERMINAL.

15. OPEN FILE EDTDMO.LAB FOR EDITING.

16. CHANGE THE LINE "EDIT OBJECTIVES" TO "RECOGNIZING A GOOD OBJECTIVE."

17. LOCATE THE STRING "WE ASSUME," AND DELETE THE COMMA.

18. LOCATE THE STRING "TOO," AND DELETE IT.

19. LOCATE THE STRING "TAHAT" AND CHANGE IT TO "THAT".

20. LOCATE THE STRING "JOIF" AND CHANGE IT TO ~~"JOB~~ "JOB IF" ".

21. LOCATE THE STRING "STAE" AND INSERT "T" BETWEEN THE "A" AND "E".

22. CHANGE THE SECOND OCCURENCE OF THE STRING "OF" TO "IN" IN THE LINE BEGINNING WITH: "ACCEPT AS EVIDENCE OF COMPETENCE...."

23. IN THE SAME LINE INSERT BETWEEN "TO" AND "THAT" THE STRING "SAY".

24. CHANGE THE STRING "ASSUNE" TO ASSUME".

25. LOCATE THE STRING "WEE" AND DELETE ONE "E".

26. LOCATE THE STRING "DON'T" IN THE SAME LINE AND DELETE IT.

27. CHANGE THE STRING "WERE'" TO "WE'RE".

28. LOCATE THE LINE BEGINNING WITH "CAN EASILE..." AND DELETE IT.

29. LOCATE THE STRING "DESIPTION" AND INSERT "CR" BETWEEN "S" AND "I".

30. IN THE SAME LINE AS STEP ~~16~~ 29, DELETE THE "O" AT THE END OF THE LINE.

31. LOCATE THE STRING "NO T" AND DELETE IT.

32. IN THE SAME LINE AS STEP 31, CHANGE "CHNAGED" TO "CHANGED" AND INSERT AN "S" BETWEEN "E" AND "I" IN THE STRING "DEIRED".

33. CHANGE ALL "TEACHING" STRINGS TO "INSTRUCTION" IN YOUR TEXT BUFFER. USE THE MACRO FACILITY.

34. CHANGE THE STRING "GOAL" TO "DESTINATION" IN YOUR BUFFER.

35. INSERT BEFORE YOUR CURRENT TEXT TWO LINES OF IDENTICAL CHARACTERS, I.E. THE % CHARACTER. USE THE SAVE AND UNSAVE COMMANDS TO REPEAT THESE TWO LINES AT THE END OF YOUR BUFFER. LIST YOUR BUFFER.

36. EXIT TO THE MONITOR.

37. OPEN THE FILE EDTDMO. TWO FOR EDITING. LIST THE FILE.

38. WRITE A MACRO TO CHANGE EACH OCCURRENCE OF "123" TO "45".

39. EXECUTE THIS MACRO TWICE.

40. LIST THE FILE.

41. MOVE THE FIRST 4 LINES TO THE BOTTOM OF THE FILE. DELETE THEM FROM THE TOP.

42. CLEAR THE MACRO AND SAVE BUFFERS.

43. WRITE A MACRO TO DELETE ALL LINES CONTAINING AN X.

44. EXECUTE THE MACRO ON THE FILE SO AS TO DELETE ALL LINES CONTAINING X.

45. DIVIDE THE FILE INTO TWO PAGES AT THE LINE INDICATED BY INSERTING FORM FEED (CTRL/L).

46. LIST THE FILE

47. EXIT THE EDITOR. REOPEN THE FILE FOR EDITING.

48. LIST THE BUFFER. LIST THE SECOND PAGE.

49. EXIT THE EDITOR.

50. COPY EDTDMO.TWO TO YOUR TERMINAL. IF BOTH PAGES ARE NOT THERE, EXPLAIN WHY NOT.

THURSDAY LAB

1. BOOT YOUR DISK. SET TIME AND DATE.

DEBUGGING TOOLS

2. ASSEMBLE/COMPILE AND LINK HELLO.MAC OR HELLO.FOR USING THE DEBUGGING TOOLS ODT OR FDT.

MACRO/LIST:TT: HELLO.MAC
LINK/DEBUG HELLO /MAP:TT:

3. SET BREAKPOINTS/PAUSES IN THE LOOPS OF THIS PROGRAM.

RUN nnnnnn;B

4. EXECUTE HELLO. EXAMINE LOCATIONS/VARIABLES AT THE BREAK POINTS.

1000;G

5. CHANGE THE VALUES OF LOCATIONS/VARIABLES AND MONITOR THE EFFECT ON THE PROGRAM.

LIBRARIES

6. COMPILE THE FOLLOWING FORTRAN SOURCE FILES INTO SEPARATE OBJECT MODULES:

FORTRAN PROG,A,B,C1,C2,D

PROG.FOR-MAIN ROUTINE
A.FOR-SUBROUTINE A
B.FOR-SUBROUTINE B
C1.FOR-SUBROUTINEC
C2.FOR-SUBROUTINEC
D.FOR-SUBROUTINED

7. CREATE AN OBJECT LIBRARY FROM THE SUBROUTINES IN A,B AND C1.

LIBRARY/CREATE NEWLIB A,B,C1

8. LIST THE SUBROUTINE NAMES FROM YOUR LIBRARY.

LIBRARY/LIST:TT: NEWLIB

9. INSERT SUBROUTINE D INTO YOUR LIBRARY.

LIBRARY NEWLIB D/UPDATE

10. LIST YOUR LIBRARY.

LIBRARY/LIST:TT: NEWLIB

11. LINK PROG.OBJ USING YOUR LIBRARY.

~~LINK PROG,C1,NC~~

LINK/LIBRARY:DK: NEWLIB
PROG,

12. RUN PROG.

RUN PROG

13. REPLACE SUBROUTINE C IN YOUR LIBRARY WITH THE NEW VERSION FROM FILE C2.OBJ.

LIBRARY/DELETE NEWLIB
C

~~LIBRARY NEWLIB C/DELETE~~,

LIBRARY NEWLIB C/UPDATE

14. LIST YOUR LIBRARY.

LIBRARY /LIST: TT: NEWLIB

15. LINK AND RUN PROG AGAIN.

LINK/LIBRARY: NEWLIB PROG
RUN PROG

16. PLACE THE SUBROUTINE IN FILE E.OBJ INTO YOUR LIBRARY.

LIBRARY NEWLIB E/UPDATE

17.A LIST YOUR LIBRARY. WHAT SUBROUTINE WAS IN E.OBJ?

17B RELINK & RUN PROG

INDIRECT COMMAND FILES

18. WRITE AN INDIRECT COMMAND FILE TO ASSEMBLE/COMPILE, LINK AND RUN HELLO.

MACRO/LIST:TT: HELLO
LINK/MAP:TT: HELLO
RUN HELLO

19. WRITE AN INDIRECT COMMAND FILE TO:

A. INITIALIZE YOUR SECONDARY DEVICE - SPECIFY VOLUME-ID.
B. COPY ANY FIVE FILES TO THAT DEVICE.
C. GET DIRECTORY OF THE SECONDARY DEVICE.

INITIALIZE/NOQUERY/
VOLUMEID DK:
DIR/BRIEF SY:
COPY

20. WRITE AN INDIRECT COMMAND FILE TO DELETE THE 1ST, 3RD AND 5TH FILES FROM THE SECONDARY DEVICE.

DIR/FULL/VOLUMEID
DK:

21. CREATE A TEXT FILE USING THE EDITOR THAT GIVES INSTRUCTIONS ON BOOTING THE FOREGROUND-BACKGROUND MONITOR USING THE BOOT COMMAND. ARRANGE IT SO THAT THIS FILE WILL BE AUTOMATICALLY PRINTED WHEN THE DISK IS BOOTED.

22. REBOOT YOUR SYSTEM TO TEST STEP 21. SET TIME AND DATE.

DELETED FILES

23. CREATE A FILE USING THE EDITOR CONTAINING YOUR NAME AND ADDRESS.

24. DELETE THE ABOVE FILE.

25. USE DIRECTORY/DELETED/OCTAL OPTION ON YOUR DISK

26. USE DUP /C SWITCH (SYS. USER GUIDE P. 8-3) TO RECOVER THE FILE DELETED ABOVE.

27. DELETE ALL FILES CREATED TODAY (/NEWFILES).

28. DELETE ALL FILES CREATED IN PREVIOUS LABS AS IN STEP #27.

FRIDAY LAB

1. BOOT YOUR DISK. SET TIME AND DATE.

XD XA XB
XC XE
ROOT

OVERLAYS

YOU HAVE ON YOUR DISK A FORTRAN MAIN PROGRAM (ROOT.FOR) AND 5 FORTRAN SUBROUTINES (XA,XB,XC,XD,XE.FOR). THE CALLING SEQUENCE FOR THESE ROUTINES IS AS FOLLOWS:

ROOT CALLS XC,XD,XE
XC CALLS XA,XD
XE CALLS XA,XB

2. LAYOUT AN OVERLAY STRUCTURE(2REGIONS) FOR THIS PROGRAM.

3. LINK THIS PROGRAM USING OVERLAYS. GET A MAP.

4. A. RUN THE PROGRAM.

   B. CHECK YOUR RESULT BY RE-LINKING ROOT WITH XA,XB,XC,XD,XE INTO ONE .SAV FILE (USE NORMAL LINK COMMAND) COMPARE MAPS OF THE TWO VERSIONS. RUN THE NEW RELINKED VERSION.

5. IF TIME PERMITS, WRITE ADDITIONAL SUBROUTINES TO BE CALLED FROM XA,XB,XD THAT WILL RESULT IN A THREE REGION OVERLAY STRUCTURE.

FOREGROUND-BACKGROUND

FOR THIS EXERCISE, YOU WILL NEED TWO RT-11 DEMONSTRATION PROGRAMS STORED ON THE DISTRIBUTION DEVICES: DEMOBG.MAC AND DEMOFG.MAC. YOU WILL HAVE TO ASSEMBLE AND LINK BOTH PROGRAMS, TAKING INTO ACCOUNT THAT DEMOFG.MAC WILL RUN IN THE FOREGROUND.

THE FOREGROUND PROGRAM DEMOFG.MAC WILL SEND A REQUEST TO THE BACKGROUND PROGRAM EVERY TWO SECONDS. UPON RECEIVING EACH REQUEST THE BACKGROUND PROGRAM WILL RING THE BELL OF THE CONSOLE TERMINAL. EACH REQUEST ISSUED BY THE FOREGROUND PROGRAM IS PUT IN A WAITING LINE (QUEUE) WHILE WAITING FOR THE BACKGROUND TO RECEIVE IT.

6. BOOT THE FOREGROUND/BACKGROUND MONITOR. SET TIME AND DATE.

7. ASSEMBLE AND LINK DEMOFG.MAC AND DEMOBG.MAC.

8. LOAD THE FOREGROUND PROGRAM CALLED DEMOFG.REL BUT DO NOT START EXECUTION. OBTAIN THE LOAD ADDRESS.

9. START THE FOREGROUND PROGRAM WITHOUT RELOADING IT.

10. KILL THE FOREGROUND PROGRAM. ↑F ↑C ↑C

11. LOAD DEMOFG.REL, USING THE SAME COMMAND AND SWITCH AS IN STEP 8, BUT ALLOCATE AN EXTRA 100 OCTAL WORDS TO IT THROUGH THE USE OF OPTION /N OF THE FRUN COMMAND. COMPARE THIS LOAD ADDRESS TO THAT OBTAINED IN STEP 8.

12. START THE FOREGROUND PROGRAM WITHOUT RELOADING.

13. RUN THE BACKGROUND PROGRAM, DEMOBG.SAV. DO YOU HEAR THE BEEPS?

14. TEMPORARILY STOP THE FOREGROUND PROGRAM. DO THE BEEPS STOP? VERIFY THIS BY RUNNING THE BACKGROUND PROGRAM.

15. RESUME THE FOREGROUND PROGRAM AND RUN THE BACKGROUND. DO THE BEEPS RESUME?

16. KILL THE FOREGROUND AND BACKGROUND JOBS.

17. REPLACE THE SOUNDING OF THE BEEP WITH THE PRINTING OF THE CHARACTER "#" BY USING EDIT TO EDIT THE PROGRAM DEMOBG.MAC. TYPE IN THE FOLLOWING COMMANDS (THE UNDERLINED MATERIAL):

```
.EDIT DEMOBG.MAC
*R$$              TAB
*G.BYTE 7$-C43$$
*EX$$
```

WHERE $ IS THE ECHO OF THE ALTMODE (ESCAPE) KEY.

18. REASSEMBLE AND RELINK THE PROGRAM DEMOBG.MAC. START THE FOREGROUND AND BACKGROUND PROGRAMS, AND THEN KILL BOTH PROGRAMS.

19. RUN THE FOREGROUND AND REPEAT STEPS 13-16.

BASIC

20. CALL OUT THE BASIC INTERPRETER.

21. ENTER THE FOLLOWING BASIC PROGRAM.

```
10   PRINT "INPUT THREE NUMBERS"
12   PRINT "I WILL PRINT THEIR PRODUCT"
15   LET P=1
20   FOR I=1 TO 3
30   INPUT J
40   P=P*J
50   NEXT I
60   PRINT "THE PRODUCT IS";P
99   END
```

22. RUN THE ABOVE PROGRAM

23. STORE THE PROGRAM ON DISK.

24. MODIFY THE PROGRAM TO PRODUCE THE SUM OF THREE NUMBERS.

25. RUN THE MODIFIED PROGRAM.

26. STORE THE MODIFIED PROGRAM ON DISK UNDER A DIFFERENT NAME.

27. CALL FROM DISK AN EXISTING PROGRAM IN A FILE MATCH.BAS.

28. RUN THIS PROGRAM. IT SHOULD BE SELF-DOCUMENTING.

29. RE-RUN THE PRODUCT PROGRAM FROM ABOVE. INTERRUPT ITS EXECUTION WITH CTRL-C AND RETURN TO THE BASIC INTERPRETER.

30. RERUN THE SUM PROGRAM.

31. RETURN TO THE RT-11 MONITOR.

digital

digital equipment corporation

Printed in U.S.A.